

**Федеральное государственное автономное образовательное
учреждение высшего образования
«Московский физико-технический институт
(национальный исследовательский университет)»**

УТВЕРЖДЕНО

**Проректор по учебной работе и
довузовской подготовке**

А.А. Воронов

	Рабочая программа дисциплины (модуля)
по дисциплине:	Практика программирования с использованием Python
по направлению:	Техническая физика
профиль подготовки:	Техническая физика космических летательных аппаратов Физтех-школа Аэрокосмических Технологий кафедра информатики и вычислительной математики
курс:	1
квалификация:	бакалавр

Семестр, формы промежуточной аттестации: 2 (весенний) - Дифференцированный зачет

Аудиторных часов: 60 всего, в том числе:

лекции: 0 час.

семинары: 0 час.

лабораторные занятия: 60 час.

Самостоятельная работа: 75 час.

Всего часов: 135, всего зач. ед.: 3

Количество контрольных работ, заданий: 1

Программу составил: Т.Ф. Хирьянов, старший преподаватель

Программа обсуждена на заседании кафедры информатики и вычислительной математики 06.02.2020

Аннотация

Курс направлен на обучение студентов основам программирования на языке Python 3. Студенты познакомятся с правилами проектирования программного обеспечения, узнают о практиках контроля качества кода. Обучающиеся освоят структурное и модульное программирование. Внимание будет уделено групповому программированию, работе в команде. В ходе курса необходимо будет выполнить курсовой проект на языке Python.

1. Цели и задачи

Цель дисциплины

Научить студентов программировать простые графические приложения на языке Python 3 как самостоятельно, так и в группе, с использованием системы контроля версий git и соблюдением принципов качества кода.

Задачи дисциплины

- Формирование у обучающихся базовых знаний о синтаксисе языка Python 3 и его возможностях;
- формирование культуры создания читабельного кода;
- формирование умения осуществлять декомпозицию проекта ПО на функции, объекты и модули;
- формирование навыка проектирования и разработки ПО с использованием системы контроля версий, в том числе в рабочей группе.

2. Перечень формируемых компетенций

Освоение дисциплины направлено на формирование следующих компетенций:

Код и наименование компетенции	Индикаторы достижения компетенции
ОПК-4 Способен решать стандартные задачи профессиональной деятельности на основе информационной и библиографической культуры с применением информационно-коммуникационных технологий и с учетом основных требований информационной безопасности	ОПК-4.2 Владеет навыками работы с компьютером и компьютерными сетями с целью получения, хранения и обработки научной информации

3. Перечень планируемых результатов обучения по дисциплине (модулю)

В результате освоения дисциплины обучающиеся должны

знать:

- Основы алгоритмического языка программирования Python 3;
- принципы создания качественного читабельного кода;
- приёмы разработки программ «сверху-вниз» и «снизу-вверх»;
- идеологию модульного и объектно-ориентированного подхода;
- типовые решения, применяемые для создания программ.

уметь:

- Разрабатывать читабельные программы на языке программирования Python 3;
- использовать как встроенную, так и доступную в Сети документацию по библиотекам Python 3;
- подключать дополнительные модули и стандартные модули и пакеты Python 3;
- создавать дополнительные модули и пакеты на Python 3 для основной программы;
- применять объектно-ориентированный подход для написания программ;
- разрабатывать программы как индивидуально, так и в команде, с использованием современных средств написания и отладки программ.

владеть:

- Одной из интегрированных сред разработки программ для языка Python 3;
- интерактивной консолью Python 3 для простых вычислений;
- основными командами системы контроля версий git;
- основным инструментарием библиотеки Tkinter;
- средствами отладки и интроспекции на языке Python 3.

4. Содержание дисциплины (модуля), структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебных занятий

4.1. Разделы дисциплины (модуля) и трудоемкости по видам учебных занятий

№	Тема (раздел) дисциплины	Трудоемкость по видам учебных занятий, включая самостоятельную работу, час.			
		Лекции	Семинары	Лаборат. работы	Самост. работа
1	Синтаксис языка Python 3			12	4
2	Проектирование ПО			8	5
3	Контроль качества кода			4	6
4	Структурное программирование			4	6
5	Модульное программирование			4	6
6	Объектно-ориентированное программирование			4	6
7	Групповая разработка программ			6	6
8	Событийно-ориентированное программирование			6	6
9	Семестровый проект			12	30
Итого часов				60	75
Подготовка к экзамену		0 час.			
Общая трудоёмкость		135 час., 3 зач.ед.			

4.2. Содержание дисциплины (модуля), структурированное по темам (разделам)

Семестр: 2 (Весенний)

1. Синтаксис языка Python 3

Интерактивный режим Python 3 и интегрированные среды разработки Условный оператор if. Вложенные и каскадные ветвления. Логические операции and, or, not. Циклы while и for. Инструкции управления циклом. Генератор прогрессий range(). Вложенные циклы. Описание функций без параметров и с параметрами. Кортежи переменных. Обмен значений. Итерируемые объекты и цикл for. Золотой фонд Python: коллекции tuple, list, set, dict. Изменяемость списка и операции с ним. Неизменяемость кортежа и операции с ним. Список кортежей. Цикл for для двух переменных. Разворачивание итерируемого объекта в параметры функции. Генераторы списков, кортежей, множеств. Ссылочная модель данных. Присваивания в Python. Интроспекция. Оператор is. Специфика Python: duck typing. Значения параметров по умолчанию. Именованные параметры функций.

2. Проектирование ПО

Проработка интерфейсов функций. Рефакторинг. Введение в ООП проектирование. Проработка интерфейсов, контрактов и ответственности классов.

3. Контроль качества кода

Читабельность кода и необходимость Style Guide. Краткая выжимка PEP8. Принцип именования переменных и функций. Документация программы

4. Структурное программирование

Инкапсуляция ответственности в функцию.
Декомпозиция. Проектирование «сверху-вниз».

Проектирование «снизу-вверх».

5. Модульное программирование

Цель и принцип разделения на модули. Создание модулей и пакетов. Возможности инструкции `import`. Проработка и документация интерфейса модуля. Локализация переменных.

6. Объектно-ориентированное программирование

Классы и объекты в Python. Создание и инициализация объекта. Инкапсуляция ответственности в класс. Принцип единственной ответственности класса. Отношения между классами: наследование, композиция, ассоциация. Диаграмма классов UML.

7. Групповая разработка программ

Каскадная модель разработки Waterfall. Итеративная разработка. Распределение ролей в проекте. Документация проекта. Необходимость контроля версий. Терминология. Система контроля версий `git`. Создание и клонирование репозитория: `git init`, `git clone`, `git status`. Контроль изменений: `git diff`, `git add`, `git commit`, `git log`, `git blame`. Ветки `git`: `git branch`, `checkout`, `merge`. Система отслеживания ошибок в проекте и управления проектом. Взаимная вычитка кода и `approve`.

8. Событийно-ориентированное программирование

Событийная модель построения приложения. Виджеты, события и обработчики событий. Свойства и упаковка виджетов. Основы библиотеки Tkinter. Создание интерактивной графической программы. Виджеты Tkinter и их упаковка в главное окно программы. Tkinter Canvas: методы, идентификаторы и теги. Переменные с обратной связью в tkinter.

9. Семестровый проект

Разработка архитектуры программного продукта. Разработка плана создания ПО. Распределение ролей участников проекта. Взаимодействие через GitHub. Коворкинг. Обсуждение с преподавателем и ментором. Сдача проекта.

5. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю)

Лекционная аудитория должна быть с меловой или маркерной доской, а также мультимедиа проектором с возможностью подключения компьютера лектора через D-SUB или HDMI. Аудитории для лабораторных работ должны быть оборудованы компьютерами по числу обучающихся с операционной системой GNU/Linux и интерпретатором Python 3, подключенными к сети Интернет. Для работы преподавателя лабораторных работ нужна также меловая или маркерная доска, а также мультимедиа проектор или большой экран с возможностью подключения компьютера преподавателя через D-SUB или HDMI.

6. Перечень рекомендуемой литературы

Основная литература

1. Лутц, М. Python [Текст] : карманный справочник / М. Лутц ; пер. с англ. И. В. Берштейна .— 5-е изд. — М : Вильямс, 2015 .— 320 с.

Дополнительная литература

1. Приемы объектно-ориентированного проектирования. Паттерны проектирования / Э. Гамма [и др.] ; [пер. с англ. А. Слинкина]. - СПб. : Питер, 2018

7. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", необходимых для освоения дисциплины (модуля)

Не используются

8. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень необходимого программного обеспечения и информационных справочных систем (при необходимости)

Для командной работы над проектами под системой контроля версий git обучающимся во время занятий необходим доступ в Интернет, в частности к сайту <https://github.com>.

Интерпретатор Python 3.

Библиотека Tkinter.

Среды разработки IDLE, PyCharm Community Edition, IDE Spyder.

9. Методические указания для обучающихся по освоению дисциплины (модуля)

Изложение материала происходит преимущественно на лекциях, сопровождается мультимедиа-презентацией с примерами кода и блок-схемами алгоритмов. На лабораторных занятиях также происходит изложение нового материала: в начале каждой лабораторной работы, по мере необходимости, а также в личных беседах тьютора с рабочими группами. На контекстах изложение нового материала исключено, преподаватель оказывает только консультативное содействие для успешного решения задач.

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)

по направлению: Техническая физика
профиль подготовки: Техническая физика космических летательных аппаратов
Физтех-школа Аэрокосмических Технологий
кафедра информатики и вычислительной математики
курс: 1
квалификация: бакалавр

Семестр, формы промежуточной аттестации: 2 (весенний) - Дифференцированный зачет

Разработчик: Т.Ф. Хирьянов, старший преподаватель

1. Компетенции, формируемые в процессе изучения дисциплины

Код и наименование компетенции	Индикаторы достижения компетенции
ОПК-4 Способен решать стандартные задачи профессиональной деятельности на основе информационной и библиографической культуры с применением информационно-коммуникационных технологий и с учетом основных требований информационной безопасности	ОПК-4.2 Владеет навыками работы с компьютером и компьютерными сетями с целью получения, хранения и обработки научной информации

2. Показатели оценивания компетенций

В результате изучения дисциплины «Практика программирования с использованием Python» обучающийся должен:

знать:

- Основы алгоритмического языка программирования Python 3;
- принципы создания качественного читабельного кода;
- приёмы разработки программ «сверху-вниз» и «снизу-вверх»;
- идеологию модульного и объектно-ориентированного подхода;
- типовые решения, применяемые для создания программ.

уметь:

- Разрабатывать читабельные программы на языке программирования Python 3;
- использовать как встроенную, так и доступную в Сети документацию по библиотекам Python 3;
- подключать дополнительные модули и стандартные модули и пакеты Python 3;
- создавать дополнительные модули и пакеты на Python 3 для основной программы;
- применять объектно-ориентированный подход для написания программ;
- разрабатывать программы как индивидуально, так и в команде, с использованием современных средств написания и отладки программ.

владеть:

- Одной из интегрированных сред разработки программ для языка Python 3;
- интерактивной консолью Python 3 для простых вычислений;
- основными командами системы контроля версий git;
- основным инструментарием библиотеки Tkinter;
- средствами отладки и интроспекции на языке Python 3.

3. Перечень типовых (примерных) вопросов, заданий, тем для подготовки к текущему контролю

4. Перечень типовых (примерных) вопросов и тем для проведения промежуточной аттестации обучающихся

1. Самостоятельное создание за короткое время в присутствии преподавателя небольших программ на Python 3.
2. Самостоятельное или групповое создание большой программы на Python 3 в публичном репозитории в течение двух-трёх недель с последующей защитой проекта.

Критерии оценивания

- 1) функциональность программы;
- 2) качество программного кода (соблюдение стиля, архитектура и общая читабельность кода);
- 3) наличие и консистентность промежуточных версий проекта;
- 4) организация групповой работы, если сдаваемая работа была групповой.

5. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности

Оценивание проводится основным преподавателем учебной группы по мере выполнения лабораторных работ студентом в течение семестра. По итогам всех лабораторных работ вычисляется средневзвешенная оценка.

Семестровая работа выносится на защиту в конце семестра на зачётной неделе, для её оценивания оперативно назначается группа из трёх преподавателей (один из которых - основной для данной группы), которая коллегиально выставляет оценку за работу.

Итоговая оценка студента по предмету является средним арифметическим между средневзвешенной оценкой по лабораторным работам и оценкой за семестровую работу.

3. Перечень типовых заданий, используемых для оценки знаний, умений, навыков
Промежуточная аттестация по дисциплине «Практика программирования с использованием Python» осуществляется в форме дифференцированного зачета. Дифференцированный зачет выставляется по результатам работы студента в течение семестра на основе оценок за лабораторные работы, и семестровый проект (семестровую работу), представляющий из себя законченную графическую программу на языке Python.

Лабораторные работы

1. Лабораторная работа "Робот"

Команды Робота для работы в интерактивном режиме.

Задачи на циклы и условия: прохождение и закрашка коридора.

Задача на вложенные циклы: сложные коридоры.

Задачи на подсчёт шагов.

Поиск выхода из лабиринта.

2. Лабораторная работа "Черепаша"

Задачи на управление исполнителем Черепаша.

Задачи на циклы: многоугольники, звезды.

Задачи на вложенные циклы.

Задачи на написание функций без параметров и с параметрами.

Задачи с использованием списков при управлении Черепашей.

Операции со списками. Генераторы списков.

Кортежи в Python. Список кортежей.

Случайное движение черепахи, модуль random.

3. Лабораторная работа "Картина"

Основные команды управления Git-репозиторием.

Задачи на построение геометрических объектов из примитивов.

Создание картины по образцу.

4. Лабораторная работа "Рефакторинг"

Структурное программирование.

Проектирование сверху-вниз.

Рефакторинг.

5. Лабораторная работа "Анимация"

Проработка интерфейсов функций.

Документация функций.

6. Лабораторная работа "Поймай шарик"

Введение в Tkinter. Событийная модель построения приложения.

Виджеты, события и обработчики событий. Свойства и упаковка виджетов.

Tkinter Canvas: методы, идентификаторы и теги.

Создание простой игры: кликать мышкой на случайно появляющихся шариках.

Виджет кнопки для перезапуска игры.

Возбуждение событий по времени.

Движение шариков с отражением от стен.

Подсчёт игровых очков.

Переменные с обратной связью в tkinter.

Ползунок, управляющий скоростью движения шариков.

7. Лабораторная работа "Пушка"

Простое объектно-ориентированное программирование. Создание и инициализация класса. Моделирование кинематики материальной точки (снаряда в поле силы тяжести).

Модульное программирование. Создание модулей и пакетов. Документация проекта.

Введение в ООП проектирование. Проработка интерфейсов и контрактов функций и ответственности классов. Отношения между классами. Наследование и композиция. Ассоциация. Диаграмма классов UML.

8. Лабораторная работа "Артиллерия"

Проектирование игры в стиле artillery game.

Командная работа.

Учимся взаимодействовать через GitHub.

Осваиваем: git branch, git merge, pull request. Механизм review.

Семестровый проект

Проект выполняется командой из 2-3 человек. Выполнение проекта в одиночку — исключение. Важно, чтобы у каждого участника проекта была чётко сформулированная часть работы, а разделение ответственности за части кода (функции, объекты или модули) происходило до написания кода.

Требования к проекту

1. Использование системы контроля версий с доступом для преподавателя.
2. Количество кода — не менее 200 строк исходного текста на Python на каждого участника.
3. Графический пользовательский интерфейс.
- 4.
5. Программа должна быть сохранять работоспособность при любых входных данных.

4. Критерии оценивания

1. Оценивается не только программа, получающаяся в результате, но и умение грамотно построить сам процесс разработки с использованием системы контроля версий и системы управления проектом на GitHub.

1. Наличие истории коммитов, в которой участвуют все участники проекта.
2. Наличие issue (bug, ticket) в системе планирования и баг-трекинга и истории их изменений (open / resolved).
3. Мелкость и логическая цельность коммитов.
4. Целостность проекта после каждого отдельного коммита (допускаются «ломающие коммиты», но не более 50%).
2. Оценивается архитектура приложения.
 1. Декомпозиция на модули по крупным разделам функциональности (модель / представление / контроллер или по-другому) и умение обосновать данный выбор.
 2. Разбиение модуля на функции небольшого размера (до 30 строк кода).
 3. Проработанность интерфейса функций и классов.
 4. Отсутствие (или минимальное и обоснованное количество) глобальных переменных.
3. Оценивается качество исходного текста программы.
 1. Код должен быть оформлен согласно PEP8.
 2. Документация функций, классов и модулей документ-строками.
 3. Осмысленные названия функций, объектов и модулей.
4. Оценивается функциональность программы.
 1. Программа должна реализовывать заявленную функциональность.
5. Оценивается наличие автоматизированных тестов, частично покрывающих функциональность программы.

За каждый пункт студент получает от 0 до максимального балла в зависимости от полноты представленного ответа (решения). Критерии проставления баллов утверждаются на заседании учебно-методической комиссии кафедры. Процент суммарно набранных баллов от максимально возможного количества определяет оценку за теоретические знания по каждому контрольному заданию:

Оценка	Набранные баллы
отлично (10)	более 88%
отлично (9)	от 78% до 88% включительно
отлично (8)	от 68% до 78% включительно
хорошо (7)	от 58% до 68% включительно
хорошо (6)	от 48% до 58% включительно
хорошо (5)	от 38% до 48% включительно
удовлетворительно (4)	от 28% до 38% включительно
удовлетворительно (3)	от 18% до 28% включительно
неудовлетворительно (2)	от 08% до 18% включительно
неудовлетворительно (1)	не более 08%

Итоговая оценка за дифференцированный зачет выставляется на основании оценок, полученных за лабораторные работы (среднее арифметическое оценок за лабораторные работы — x), и оценки за семестровый проект (далее y) по следующей формуле $(x+y)/2$, с округлением результата до целого сверху.

5. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности

Презентация проекта занимает время от 15 до 30 минут и проводится на зачётной неделе или за неделю до неё (при досрочном завершении выполнения проекта группой студентов).

Каждый из участников группы участвует в представлении проекта.

Мультимедиа-презентация должна содержать

- 1.** слайды, отражающие архитектуру программного продукта.
- 2.** слайды, отражающие распределение ролей в команде.
- 3.** слайды, отражающие методику разработки и план разработки продукта.